Universitat de Lleida

TREBALL FINAL DE MÀSTER

Estudiant: **Dídac Florensa Cazorla**

Titulació: Màster en Enginyeria Informàtica

Títol de Treball Final de Màster: eCut, a new cloud-based decision support system to improve productivity in meat-packing plants. Based on a microservice architecture to handle executions in OpenNebula environments.

Director/a: Jordi Mateo Fornés i Lluís Miquel Plà Aragonés

Presentació

Mes: Juliol

Any: 2019

eCut, a new cloud-based decision support system to improve productivity in meat-packing plants. Based on a microservice architecture to handle executions in OpenNebula environments.

Author: Dídac Florensa Cazorla

***Abstract*—The pig farming sector is a very competitive business where innovation is a key factor. On the research side, there exists a large body of models and research that too often do no reach real companies for practical use. In this paper, we propose a cloud-based decision support system to overcome the main practical limitations and make a contribution in a specific decision problem of pig meat-packing plants. We propose a tool based on a scalable cloud architecture based on microservices and mainly focused at: (i) integrate the data in the system, (ii) integrate the model, (iii) automate the resolution process, and finally present the results in an interactive way to the end-users. The results show the benefits of the decision support system assisting a real Spanish meat-packing plant to make a better negotiation, planning, product profitability studies or marketing campaign. It is demonstrated the competitive advantages of using the proposed model in a usable, flexible and transparent way.**

***Index Terms*—Cloud computing, Software as a Service, Meat-Packing plant, Optimization models, Agrobusiness, Virtualization**

I. Introduction

The development of decision support tools for agriculture in general, and in the pig sector in particular, has been the subject of research for several decades. Big pig companies have allocated during the last years, large quantities of resources and money to optimize their process and, in the future, lowering the unitary production cost. The majority of the decision tools described in the literature are based on simulation and optimization models. Nevertheless, as it has been observed by several authors [1], [2], there are several reasons to explain why the majority of research tools does not impact on the society. The main reasons can be summarized as a) unclear focus of the project, b) steep learning curve, c) lack of interface with the current management information systems, or d) hard system maintenance. These works expose the difficulties for the agribusiness to find easily accessible and usable models.

The development of applications as Software-as-a-service (SAAS) hosted in a cloud eliminate the requirement for having a local powerful computer with an environment dedicated to solving sophisticated complex prescriptive and predictive models. The only thing required to the user is an electronic device connected to the Internet. Thus, with only a web browser, decision makers obtain access to a service with almost unlimited computing power, no matter which device is used (hand-handled device, desktop computer or laptops).

The cloud provides elasticity and is aimed at allowing resources as pay-as-you-go and pay-per-use. Cloud elasticity is a vital property to improve the quality of the services [3] and saving costs. The most important provider is Amazon, concretely AWS, see [4], that permits configure and deploy virtual machines using a pay-as-you-go. Cloud providers can add or remove features without interruption to adapt load and traffic variation on demand. This way, a cloud-based decision support system is ready to fit unpredictable demand and planned business growth for different users at a time. Close to cloud elasticity appears a new term that permits deploy a machine with the resources that the process needs, this term is Virtualization [5]. The idea of using virtual machines to solve complex tasks in a decision support system is to create a decoupled architecture and exploit cloud elasticity.

Nowadays, the monolithic architecture [6] is present in a lot of systems with we are working. In the past, this architecture was simple to develop because we used only one system for all the services. The inconvenient is that they are hard to solve. For example, is hard to redeploy when there are updates, is not scalable, is and also is difficult to apply to new technologies. So, this document presents another type of architecture to avoid these problems and overcome the problem of developing a tool hard to maintain and keep update, concretely, a decoupled architecture based on microservices.

In the meat-packing plants context there is a gap in the literature related to practical models and decision support systems. The complexity of cutting and processing meat is related to the trend of zero stock since meat is a perishable product. This means all carcasses arriving to the plant are cut, packed and delivered the same day with a low storage capacity. Authors in [7] describe a linear programming model to schedule the tasks in a beef packing plant. Furthermore, in [8], the authors propose a simulation model to simulate the activity of a medium size Spanish pig meat packing-plant. The idea behind this work was helping decision-makers to understand how their system behave and to compare their performance under different conditions to increase both productivity and reactivity. The simulation model represented a practical approach for comparing different production plans of a pig meat packing plant.

Nevertheless, a simulation model is useful to explore and value alternatives, but not enough to assist in tactical and operational decisions in detecting the best ones. This way, we propose the development of an optimization model inspired in the simulation model of [8] to optimise the portfolio of products maximising the carcass value and so, improving the production plan and the task of traders in a meat-packing plant. We propose a linear programming model that takes advantage of the optimal cutting of a carcass, reports the

shadow prices, and the reduced costs in view of assisting plant production planners and commercials in their daily activity of price negotiations with clients, or in the development of marketing campaigns.

The main contribution of this work is to provide a decoupled decision support system that takes advantage of cloud elasticity and assists commercials in a meat-packing plant. Therefore, we propose:

- A microservice to handle the virtual resources and model executions.
- An scalable decoupled architecture to orchestrate the microservices to assist the decision-making process.
- A decision support system to manage the data related with the meat-packing plant, to hide the complexity of launching a mathematical model, to automatize the execution and also to provide useful information to assist commercials in this context.

The remainder of the paper is organized as follows, section II presents the Decision Support System implemented, the new microservice to handle executions and the architecture that orchestrates the decision support system and also integrates the optimization with the cloud computing features is proposed. Further on, on the section III the problem which solves the optimization model, a discussion of the results and some practical implications. Finally, in section IV outlines the main conclusions and future work.

II. Methodology

A. Pig meat-packing plant

The pig meat packing industry handles the slaughtering, processing, packaging, and distribution of meat from pigs. Meat-packing plants can be located besides or apart an abattoir.

These plants can receive carcasses each day from different abattoirs. The sale's department manage the orders issued by customers and production planner decide and sort which orders to serve planning the production of a day for the next one. According to pending or arriving orders production can be re-planned and sale's prices adapted to the moment priorities and/or clients. The sellers are constantly attending clients and negotiating price and delivery conditions of final products with customers. Quite often, sales and production departments operates without a fluent communication between them and tends to be unidirectional from sales to production. In addition, they use rudimentary tools to accept commercial agreements and plan their production. Sellers use their know-how and expertise to negotiate the prices with clients. While the customer tries to lower the purchase price, the seller has to assure a minimum benefit while keeping the customer satisfied. Therefore, the uncertainty on a fair price and conditions besides the risk of incurring in planning mistakes makes difficult the negotiation.

The main elements of the plant operation are the cutting tree, defining the set of products, and the type of animals to process (breed, lean content and weight). The most important constraint in production is the number of cutting lines that limit the processing capacity in terms of carcasses per hour, per day or per week. Moreover, the plant design, the layout, the number of automated tasks, the number of available machines either for cutting or measuring weight, size or lean content and the personnel may also impose additional constraints. While a simulation model can consider most of the particularities of the plant, the optimization model can get the optimal production cutting plan considering most the operative constraints of the daily activity of the plant.

B. Design of the Decision Support System

The Decision Support System (DSS) proposed in this work aims to automate all the steps required to deal with a mathematical model aimed to optimize the production of a meat-packing plant such as: data integration, data processing, model execution, and results synthesis and also to hide all the complexity related to these steps.

Therefore, the main features of the decision support system designed can be summarized as:

- **Data integration:** The DSS proposed can be fed with private data from a meat-packing pant, coming from Excel, CSV files or by HTML forms. The system needs information about the list of products developed in the plant, the cutting tree leading to the product portfolio, the current list of prices and also the type of carcasses (breed, lean content and weight).
- **Data processing:** Before launching the model a pre-processing step needs to be done to retrieve data from the database and prepare the parameters to feed into the mathematical model. After solving the model, a post-processing step needs to be done to analyze the solution and perform a synthesis of the results.
- **Optimal production:** From the optimal solution of the model, the optimal cutting of a carcass is derived. Hence, the portfolio of products extracting more value from the carcass is obtained with corresponding shadow prices and reduced cost to guide and assist the decision maker.

C. Implementation of the Decision Support System

The main part of this implementation is the **Dashboard**, see the figure 5. The Dashboard is an important element of the interface with the user who can interact with all the features necessary to run the model and revise the results. The input parameters like the set of products and the cutting tree are incorporated by the user using different forms (figure 5, Product box and Cutting trees box). The system has automated the **Data Processing** to solve the mathematical model by a **Data Integration** that makes available the input data of each user. So, there is another form to launch a new execution with the input parameters selected by user. This form is implemented in three steps. The first to choose which model the user wants to run. Next, to choose which operation wants to launch according to the model selected in the first step. And finally, is to enter the requirements needs to execute correctly the model, in this case, the prices of every product.

Another feature is the **Optimal production** where the commercials can evaluate the results obtained of the model

execution. In this case, the system show three tables, one to represent a generic product, another to all the sub-products and finally, the product family of the carcass. A part, includes the possibility to download these results into a CSV file.

D. Architecture of the Decision Support System

This subsection presents the cloud architecture that provides a platform to offer knowledge as a service to the meat-packing plants professionals. This architecture orchestrates the decision support system. The architecture is composed of three layers: the frontend that represents the presentation layer, the backend and the microservices that represents the business logic layer, and finally, the data layer that represents the database. This separation makes this architecture flexible scalable and hybrid.

The architecture is built under the platform OpenNebula [9]. The main reason for choosing OpenNebula was to take advantage of the Stormy server. This is a private cloud deployed with the OpenNebula platform that belongs to the University of Lleida[10]. Nevertheless, the architecture is capable to be deployed or integrated into any commercial cloud, such as Microsoft Azure[11] or Amazon Cloud[4]. All the parts of the cloud-based service were deployed on this platform using virtual instances of Centos7 images as the operating system.

Figure 1 shows the layers' skeleton. The architecture was designed to be flexible, elastic, scalable, easy to maintain and multi-purpose. The main function of the data layer is storing the information using different heterogeneous sources, such as a none relational database (MongoDB). The presentation layer is aimed to handle the interaction with the user, gathering and displaying all the information. Finally, the business logic layer represents the execution of mathematical models, the data (pre and post) processing and the communication between the data and the presentation layer.

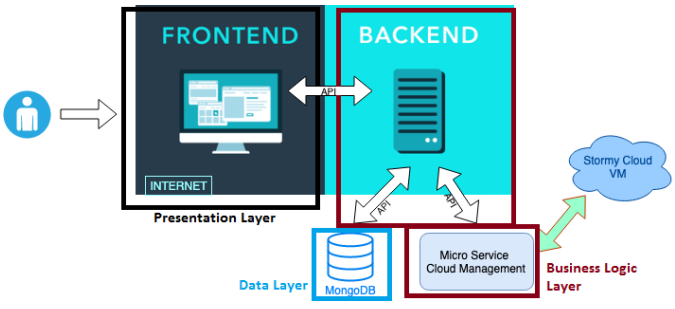


Fig. 1. Schema of layer architecture proposed, highlighting the principal components developed.

The DSS is merged with a cloud computing to avoid the barriers of software installation and maintenance inside the companies computers. Moreover, a cloud-based service can perform software updates seamlessly and can deliver a generic product that can be used by everyone independently of the knowledge in data-science. Therefore, the DSS proposed in this work is designed as a Software-as-a-Service (SAAS). The SAAS allows users connect to cloud-based applications over the Internet. So, this service offers features like the high vertical scalability and access to the application from anywhere.

1) Data layer: The data layer stores all the information related to the users of the application, the model which the users can launch, the products and cutting trees, and the executions with their results. Thus, in this case, the system has one none relational database to save all the information needed to run a complete execution and obtain the results correctly. The database is implemented with MongoDB [12] because is an open source software for the creation and management of document-oriented, scalable and high-performance database. This manages collections of documents similar to the JSON[13] data format. MongoDB is the best system to represent the NoSQL [14] databases because, nowadays, it is the main system chosen to implement databases. The data model is designed to save all the information by a meat-packing plant and by the user. The meat-packing plant save all the products and the cutting trees of each one and for a concrete plant. By the user, the database saves all the information about the executions: the input parameters, the output parameters and all the information related with the virtual machines that run the model.

2) Presentation layer: Secondly, the presentation layer is the visible part of the application. This represents the tool that the decision makers see and which interacts with. The presentation layer is a web application accessible from any modern browser and any device connected to the Internet. This is a soft layer that does not require too many computational resources. This layer is executed on the users' devices. Thus, the only requirement to use the system is a device with an active internet connection. The user interface of the presentation layer makes the interaction between the business logic, the data-layer and the decision-makers as easy, transparent and usable as possible to hide and automate all the complex steps.

The presentation layer has been implemented using the AngularJS [15] structural framework. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJS is based on a Javascript framework and it used TypeScript which facilitates the implementation of the code. A part, it offers stability during a long period, so that, we can continue deploying the presentation layer with new features and improve the quality of this.

3) Business logic layer: Finally, the core that makes everything work is the business logic. The business logic is the core of the application and handles the communication between the data and the presentation layer.

The Backend component provides all the information needed by the client, so it can display it in a user-friendly way, and also receive all the information provided by the user and save it or transform it as needed. It was implemented in Javascript[16], concretely using the framework NodeJS[17] the framework express and Gulp[18], these tools allow programmers to scale the applications in horizontal as well as the vertical directions and also to manage and automate the tasks in Javascript.

Furthermore, the backend has to provide communication between the database through REST API calls. To do it, we both use the mongoose framework to develop the data models and gulp to map API calls to javascript methods which can receive input and generate a response to deal with users requests.

Another essential feature of the backend is to handle the executions. This way, the backend uses the cloud management microservice to create, execute, and destroy on demand the task required by the users. Moreover, the tasks are automatized using python scripts that are focused on retrieving the data from the database using pymongo, prepare the data to feed the mathematical model (pre-processing), launch the mathematical model using pyomo and clean the output to be stored again to the database (post-processing). The figure 2 shows the tasks into a virtual machine.

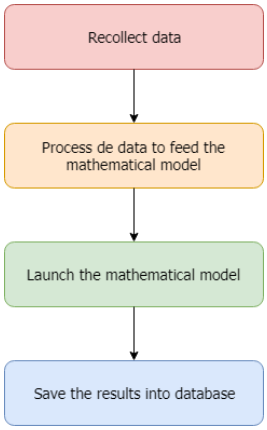


Fig. 2. Diagram of automated task using python to be solve into the virtual machines

The figure 3 shows the life cycle of an execution a when the server launch a new execution. As we can see, first of all, the server create a machine using the api-rest commented in the section II, then, create the new execution in the database, execute the model and save the results into the database while computing node starts the process to destroy the virtual machine.

The business logic receives all the data from the presentation layer, transforms and manipulates this data and ensures that this data is stored in the data layer. On the one hand, the Backend component can be considered the brain that orchestrates all the skeleton. On the other hand, the Microservice cloud management can be understood as the heart and the lungs, responsible for solving the hard computational tasks. The main characteristic of these virtual machines is volatility and elasticity. It means that virtual machines are dynamically created and destroyed, on demand. To sum up, this is the layer where the work is done. This is the central part of the system.

E. Workflow of the Decision Support System

Therefore, a sample DSS workflow can be the next: first of all, the commercial needs to define all the products and all the cutting trees of every product to after, run the model.

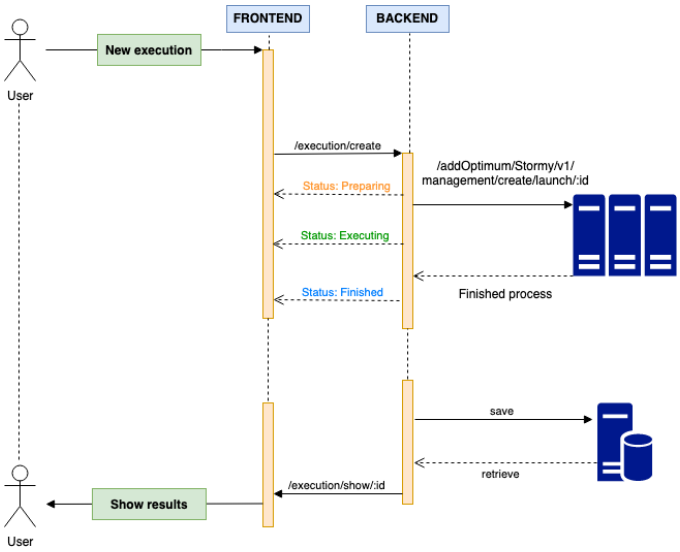


Fig. 3. Life cycle of an execution inside the DSS. It highlightes the interaction between all the componentents described.

In the cutting trees is mandatory define the % of the weight that generate. Once we are defined correctly the samples, the commercial can create a new execution inside the DSS to start using it. The next step is launch the model, so there is a step-by-step form allows the user introduce the prices and start the optimization. And, once the process has finished, the user can consult the three tables generated by the model execution. The main information delivered by the tool is the units that the company should have to produce by the defined prices to obtain benefits. A part of that, it shows the reduced cost and the dual price to help on the decision-making process. Figure 4 depicts the overall process.

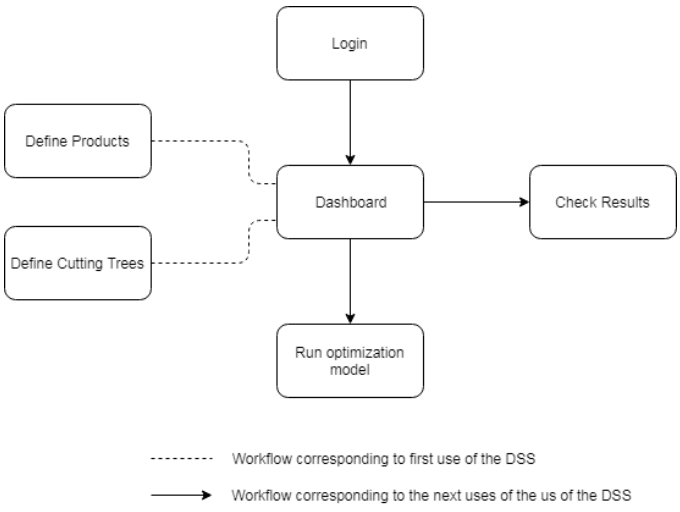


Fig. 4. Workflow within the DSS. Dashed lines depicts the steps required the first time the DSS is used. Normal line, depicts all the operations the users can realized inside the DSS.

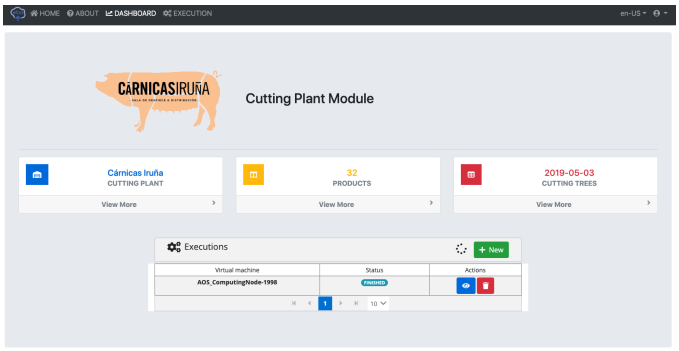


Fig. 5. Dashboard

F. Microservice to handle exectuions

This sub-section presents the implementation of a microservice to handle executions in a virtual machine under the cloud platform OpenNebula. This service was also designed to provide vertical and horizontal elasticity. Moreover, this is a decoupled generic service that can be called from any application. This microservice was implemented as a REST system, consult this document [19] to know which are the many advantages of using this type of system.

The API-rest system is implemented in Java [20] language because it is an object-oriented language which is perfect for managing the virtual machines. We used Spring because is framework based on MVC (Model View Controller) controller and also presents templates for various technologies among which we can highlight the following: JDBC, Hibernate and JPA. Finally, OpenNebula provides a OpenNebula Cloud API Specification for Java that was designed as a wrapper for the XML-RPC methods, with some basic helpers.

The service controls the cloud quota to instantiate tasks in virtual machine. When the system is saturated, and there are not enough virtual resources for proper execution of the task, then the task keeps in the queue until the system has enough resources to deal with it. Task queue aims at dispatching task requests to virtual machines. Tasks are dispatched in a safe, steady rate and with a guarantee. Hence, the system is providing a reliable service by ensuring the performance of all the tasks and also a quality service by fitting the virtual resources to each system state.

This service provide tools to perform operations such as instantiate a new virtual machine, destroy a virtual machine or rescale a virtual machine. Moreover, using SSH protocol the service can communicate and transfer data between the virtual machines and perform remote executions on them.

The figure 6 represents a typical possible situation that an API-rest can make. This way is the most usual situation because the system creates a machine, launch execution, save or return the results and finally, destroy the machine, so that the machine liberates resources used for nearby virtual machines, and only using one request to build the described process.

Applied to eCut DSS, this microservice allows creating a virtual machine from a template with all the required software

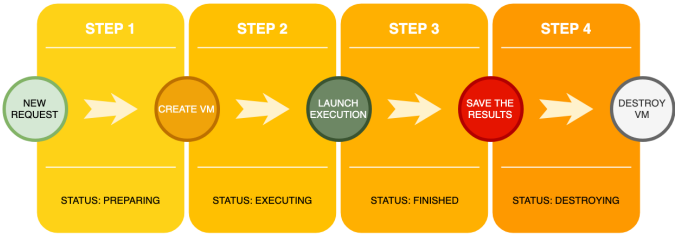


Fig. 6. Possible process using api-rest

preinstalled, launch the execution of the model (with the pre and post processing scripts) using SSH protocol and finally delete the virtual resources.

III. Case Study

A. Parameters of the company

The company "Carnicas Iruña SA" (former Carnicas Iruña-Velasco) settled in Orcoyen (Navarre, Spain) provided the data, collaboration and support for this case study. The original framework was a collaborative project understood as a joint collaboration between the University of Lleida and Carnicas Iruña to improve the knowledge in this kind of processes. In this case study, we use the cutting tree patterns and the products elaborated in this company. And, as we said in the subsection II-A the cutting tree patterns are defined by organized by the size and breed. In this company works with three types of size (Small, Medium and Big) and two types of breed (White and Pietrain), but the first version of that model which we are presenting in this document are ready to optimize for the size **Medium** and for the breed **Pietrain**.

The figure 7 shows the products considered in the optimization model. To know more about all the cutting trees of the products, consult the document [8] where it specifies the cutting trees.

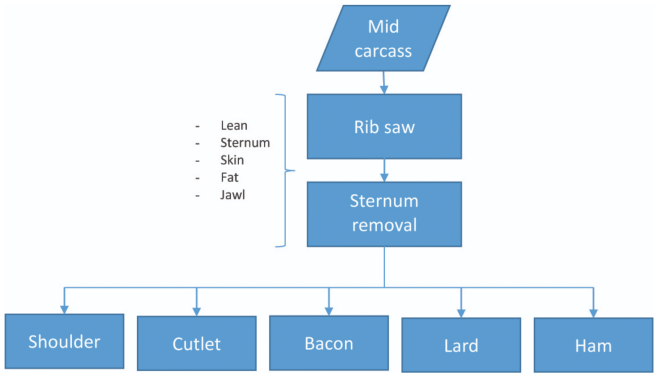


Fig. 7. Meat cutting tree

B. Results and discussion

Before start commenting the results, we can remember what the mathematical model tries to offer. So, in this case, the function tries to optimize the production of any kind of

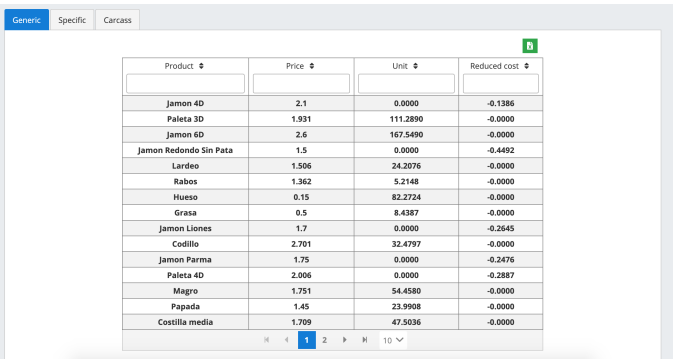
products extracted of a pig carcass according to the tree cutting patterns and the sale price. In other words, the model gives how many units has to produce of every product to obtain the maximum benefits, and a part of that, according to the table which the user is analyzing, gives the reduced costs and the dual prices.

To obtain these results, we used the DSS proposed in this work to integrate the input data of Carnicas Iruña and then run the optimization model.

The figure 8 shows the optimal solution. Looking this table, we can observe that the units equal to 0 have reduced cost different to 0. It means that, the reduced cost of a variable in the linear programming model is the amount by which the coefficient of that variable in the objective function must change so that the optimal solution produce this kind of product.

For example, in ***Jamon 4D*** product, the function solves that is not necessary produce units about this type of product, according to the other products, but the reduced cost solves with a -0.1386 value. In conclusion, we can interpret this number like the necessary value which the user has to increment because the ***Jamon 4D*** should be part of the objective function and so you can generate units.

In other words, if the user increments the price 0.1386, the model will solve how units have to produce to obtain benefits. On other hand, probably there is a product that in this case does not occur. And this conclusion can be applied for every product of the table without units. So, the commercial knows that if he wants the production to be optimum and the product would be profitability, he has to increase the price. Or, in other vision, it helps to know him the interval price that he can negotiate.

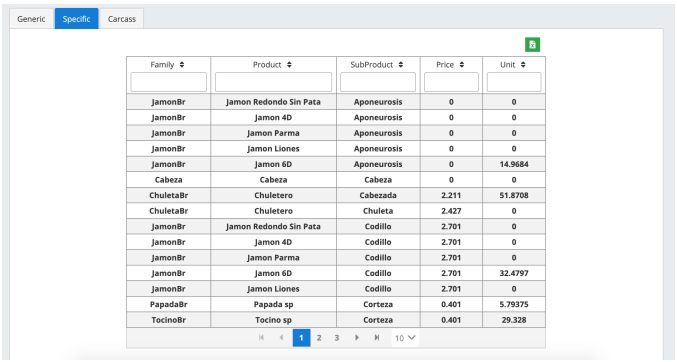

Generic Specific Carcass

Product	Price	Unit	Reduced cost
Jamon 4D	2.1	0.0000	-0.1386
Paleta 3D	1.931	111.2890	-0.0000
Jamon 6D	2.6	167.5490	-0.0000
Jamon Redondo Sin Pata	1.5	0.0000	-0.4492
Lardeo	1.506	24.2076	-0.0000
Rabos	1.362	5.2148	-0.0000
Hueso	0.15	82.2724	-0.0000
Grasa	0.5	8.4387	-0.0000
Jamon Liones	1.7	0.0000	-0.2645
Codillo	2.701	32.4797	-0.0000
Jamon Parma	1.75	0.0000	-0.2476
Paleta 4D	2.006	0.0000	-0.2887
Magro	1.751	54.4580	-0.0000
Papada	1.45	23.9908	-0.0000
Costilla media	1.709	47.5036	-0.0000

1 2 10

Fig. 8. Generic table

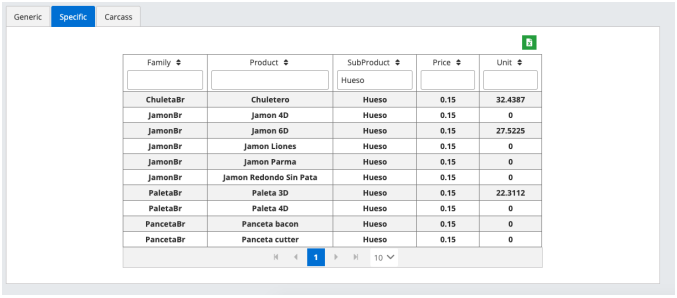
The next table, see the figure 9, is the specific table shows the complete cutting tree by family, product and sub-product. Therefore, in this case there is a list of which sub-products are being generated, a part of its product and its family. These families are the different cuts of the carcass, this is how we will see in the carcass table. So, analyzing this table, the user can look that how many units have to produce of a concrete sub-product and product. Or another way of analyze this table is complementing the generic table to know which is the product that generate this type of sub-product. For example, taking a real product of the generic table, in this case ***Hueso*** which the model solves with 82.2724 units, and the user decides to know which products contain ***Hueso*** in their solution. So, to know this information, the user has to move into the specific table, search this product in the sub-products and the list returns the solution. As the figure 10 shows, ***Hueso*** has been produced for ***Paleta 3D***, ***Jamon 6D*** and ***Chuletero***. And if we sum the units about these three products the results is the 82.2724 units, the same units of generic table. With this information, the commercial knows which products are producing each type of sub-product.


Generic Specific Carcass

Family	Product	SubProduct	Price	Unit
JamonBr	Jamon Redondo Sin Pata	Aponeurosis	0	0
JamonBr	Jamon 4D	Aponeurosis	0	0
JamonBr	Jamon Parma	Aponeurosis	0	0
JamonBr	Jamon Liones	Aponeurosis	0	0
JamonBr	Jamon 6D	Aponeurosis	0	14.9684
Cabeza	Cabeza	Cabeza	0	0
ChuletaBr	Chuletero	Cabezada	2.211	51.8708
ChuletaBr	Chuletero	Chuleta	2.427	0
JamonBr	Jamon Redondo Sin Pata	Codillo	2.701	0
JamonBr	Jamon 4D	Codillo	2.701	0
JamonBr	Jamon Parma	Codillo	2.701	0
JamonBr	Jamon 6D	Codillo	2.701	32.4797
JamonBr	Jamon Liones	Codillo	2.701	0
PapadaBr	Papada sp	Corteza	0.401	5.79375
TocinoBr	Tocino sp	Corteza	0.401	29.328

1 2 3 10

Fig. 9. Specific table


Generic Specific Carcass

Family	Product	SubProduct	Price	Unit
		Hueso		
ChuletaBr	Chuletero	Hueso	0.15	32.4387
JamonBr	Jamon 4D	Hueso	0.15	0
JamonBr	Jamon 6D	Hueso	0.15	27.5225
JamonBr	Jamon Liones	Hueso	0.15	0
JamonBr	Jamon Parma	Hueso	0.15	0
JamonBr	Jamon Redondo Sin Pata	Hueso	0.15	0
PaletaBr	Paleta 3D	Hueso	0.15	22.3112
PaletaBr	Paleta 4D	Hueso	0.15	0
PancetaBr	Panceta bacon	Hueso	0.15	0
PancetaBr	Panceta cutter	Hueso	0.15	0

1 10

Fig. 10. Specific table - ”Hueso” example

And finally, the figure 11 represents the final results of the carcass's products. As can see in this table, there are three columns which represent the ***Primary cut***, the ***Dual price*** and ***% of cutting tree***. The first column are the first cuts that can be extracted from the carcass, and during the previous analysis or in the application are known as the families of the products. And, a part of that, they take part of the restrictions defined for the function. So, as we can look at the table, the important column is the dual price because it contributes value to the analysis and to take good decisions. The ***% of cutting tree*** column gives us information about what we sent as input data to the model.

In this case we are maximizing, the improvement will mean an increase of the optimal value. The table of the figure 11 has dual price according to the primary cuts of the carcass, and as it can see, there are some equals to 0. It means that

even if the initial price of this particular case is increased, it will not have an effect on the objective function nor on the final optimal value. But, there are some primary cuts with dual prices different from 0 give us important value to help the decision-making. This price tells us how much you could earn if you increase the coefficient of the restriction. So, to apply this interpretation in the results that the figure 11 presents, we will use the family ***JamonBr*** to analyse its dual price.

The family ***JamonBr*** has a dual price of **1.84** about the restriction that represents this product family. So, according to the interpretation we have given earlier about the dual price, we can conclude that this 1.84 is what we would win if we could produce a further unit of this product. This information helps the commercial take better decisions because s/he knows how much money can enter if produces any product of this family. Thus, it helps to evaluate if compensate to produce this product taking into account everything that can lead. So, in this case we have interpreted the family ***JamonBr*** but it is the same for all the other families.

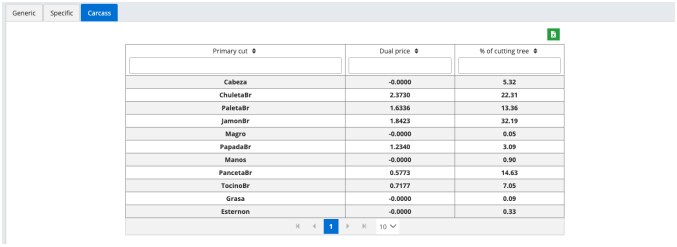


Fig. 11. Carcass table

There is overwhelming evidence explained in this study to support the adoption of this system for the meat-packing plants community. Furthermore, the economic benefits and the simplicity makes this system powerful and indicated to resolve real life problems and help farmers to take better strategical, tactical and operational decisions.

C. Practical implications

During the before subsections we have presented the results and a discussion about them. We are focusing on the price negotiation but there more applications that incites to make. The price negotiation is, probably, one of the most important implications where the commercial works, but, is important create a good marketing campaign to take one's fancy. As we said during the discussion, the prices results helps to the commercial negotiates the sale price with a costumer because with this information, knows the interval that the production of a concrete product is beneficial for the company.

Also, to design a good and productive marketing campaign, the professionals need about which prices can based it. The importance of any marketing campaign is crucial because the main idea is to attach the maximum number of costumers. A part of that, is important keep the message that the company is re-transmitting and to keep customers and what you offered in the campaign is the same that applies to them. So, to design this campaign, a previous study is needed to be able to implement the perfect campaign and that best suits the company. To help this study, the commercial can use the tables presented before and cross the results with previous executions that show price variations and cutting trees values.

The system stores the historical execution fact that allows us to analyze previous results and consult the variation of a concrete product, sub-product or family, so the commercial can calculates the profitability of any product. As we know, the profitability of a product represents the ability to generate sufficient profit, so a product will be profitability if it can generate higher revenues than you spend. Thus, with the historical results, it permits calculated the profitability of a product during a concrete period to analyze if is necessary to make a decision about it.

Finally, the system can be utilized in daily operations to provide substantial cost savings and improve productivity. The reduction of the complexity to obtain the optimal production by developing friendly graphical interfaces and automating all the tasks decrease the steep learning curve and make the tool easy to keep using by the commercials. Besides, the architecture based on cloud and microservices helps the researcher to upgrade, scale, and extend current features. As a lesson learned, the decision support system needs to fit the decision maker needs and be as simple as possible proving smart information to guide and assist their decisions.

IV. Conclusion and Future work

The novelty of this research is the development of a usable, flexible and scalable DSS to support the decision making process applied in the meat-packing plants context. A part, include the development of a system prepared to deploy more than one model and new api-rest to manage new virtual machines. Moreover, the capabilities of automation and integration are the key-factors that differentiate this service from others. Thus, the services presented in this article are a powerful seed for a much bigger services with a great potential to become reference in the agribusiness world and the deployment of different types of models.

Merging the potential of cloud computing and optimization models with the usability and portability makes the users lie easier and comfortable. This way, the process of making strategical, tactical and operational decision becomes easy. In conclusion, this system transform a manual process to detect which products have to produce according to the sale price to an automation of this calculation avoiding any human mistake.

Regarding the future work, is important to highlight the ability of the system to deploy different operations and also new models with their features. Implements and deploy new option to test the scalability of the application. Implements new requests to manage the virtual machines even thought the api-rest doesn't create them. Less important for the decision support system but related with the usability and the comfort of the service it will be crucial the implementation of a deeper notification service. Finally, implements new views about comparison the results or to evaluates the prices during a concrete period. Or, apply benchmarking to compare the

results. And, to conclude, the next step to improve the DSS is try to explore synopsis between the results of the simulation model and optimization model.

REFERENCES

1 J. Kamp, "Knowledge based systems: from research to practical application: pitfalls and critical success factors," vol. 22, no. 2-3, pp. 243–250. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168169999000320

2 B. Manos, K. Paparrizos, N. Matsatsinis, and J. Papathanasiou, *Decision support systems in agriculture, food and the environment: Trends, applications and advances*.

3 Y. Al-Dhuraibi, F. Paraiso, N. Djarallah, and P. Merle, "Decision support tools for agriculture: Towards effective designand delivery," *IEEE*, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7937885

4 "Amazon AWS," https://aws.amazon.com/, online; Accessed: May, 2019.

5 N. Saswade, D. V. Bharadi, and Y. Zanzane, "Virtual machine monitoringin cloud computing," *7th International Conference on Communication, Computing and Virtualization 2016*, 2016. [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S1877050916001496

6 M. Villamizar, O. Garcés, H. Castro, M. Verano, L. Salamanca, R. Casallas, and S. Gil, "Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud," *IEEE*, 2015. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7333476

7 A. Bixby, B. Downs, and M. Self, "A scheduling and capable-to-promise application for swift & company," *Interfaces*, vol. 36, pp. 69–86, 02 2006.

8 L. M. P. Aragonés, A. P. Bernaus, E. N. Roig, J. M. Fornés, P. Tarrafeta, D. Mendiorozm, L. P. Cànovas, and S. L. Nogales, "Economic Assessment of Pig Meat Processing and Cutting Production by Simulation," *International Journal of Food Engineering.*, 2018. [Online]. Available: https://www.degruyter.com/view/j/ijfe.ahead-of-print/ijfe-2018-0100/ijfe-2018-0100.xml

9 "OpenNebula," http://opennebula.org/, online; Accessed: May, 2019.

10 "Universitat de Lleida," http://udl.cat/, online; Accessed: May, 2019.

11 "Microsoft Azure," https://azure.microsoft.com/, online; Accessed: May, 2019.

12 "MongoDB," https://www.mongodb.com/, online; Accessed: June, 2019.

13 "JSON," https://www.json.org/, online; Accessed: May, 2019.

14 "NoSQL," https://www.mongodb.com/nosql-inline, online; Accessed: June, 2019.

15 "AngularJS — Superheroic JavaScript MVW Framework," https://angularjs.org/, online; Accessed: May, 2019.

16 "JavaScript," https://www.javascript.com/, online; Accessed: June, 2019.

17 "NodeJS," https://nodejs.org/, online; Accessed: June, 2019.

18 "Gulp," https://gulpjs.com/, online; Accessed: June, 2019.

19 R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," *University of California*, 2000. [Online]. Available: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

20 "Java," https://www.java.com/, online; Accessed: May, 2019.